KOREAN PATENT ABSTRACTS(KR)

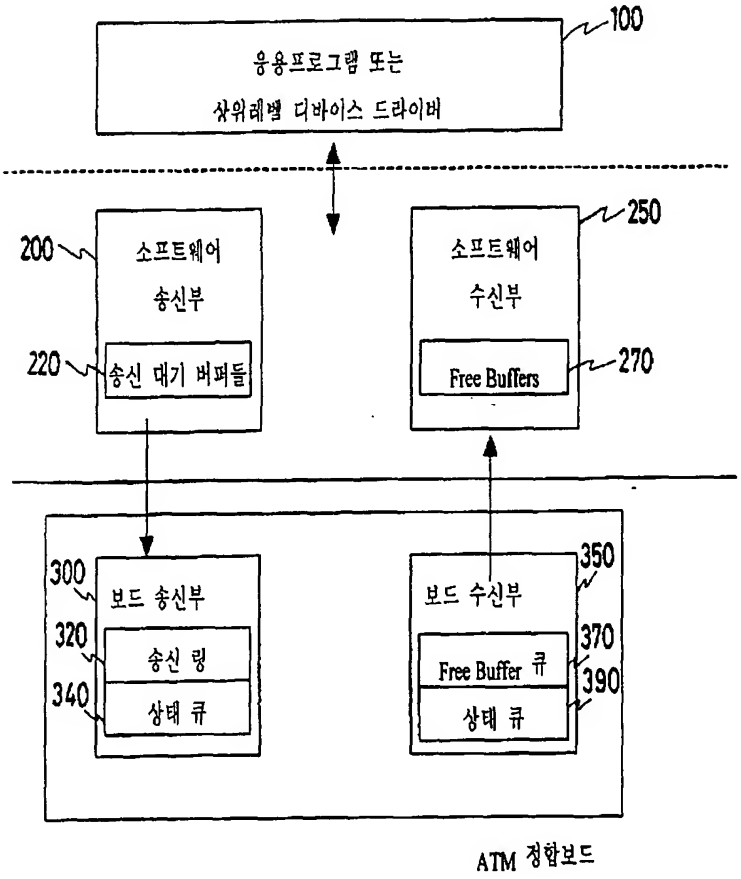
Document Code:A
(11) Publication No.1020000047035 (43) Publication.Date. 20000725
(21) Application No.1019980063784 (22) Application Date. 19981231
(51) IPC Code:
H04L 12/28

(71) Applicant:
KOREA TELECOM
(72) Inventor:
SUL, GEUN SEOK
LEE, HAE YEONG
JUNG, SANG HYEON
JUNG, HAK JIN

(30) Priority:

(54) Title of Invention
BUFFER CONTROL METHOD FOR USER DATA TRANSMISSION AND RECEPTION

Representative drawing



(57) Abstract:

PURPOSE: A buffer control method for user data transmission and reception is provided to efficiently transmit and receive user data in case of supporting the adaptive layer-5 in an ATM board, so as to be used for other adaptive layers not requesting real-time transmission.

CONSTITUTION: An adaptive layer-5 CPCS PDU is provided with an ATM interface board connected to an application program or high level device driver(100) with a software transmitting unit(200) and a software receiving unit(250) in the center. The ATM interface board composed of a board transmitting unit(300) and a board receiving unit(350) exchanges data signals with the

high level device driver(100) through the software transmitting unit(200) and the software receiving unit(250). The board transmitting unit(300) comprising a transmitting ring(320) and a status queue(340) provides an adaptive layer-5 CPCS PDU segmentation function. The board receiving unit(350) provides an adaptive layer-5 CPCS PDU reassembly function.

if display of image is failed, press (F5)

(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(51) Int. Cl.		(11) 공개번호	특2000-0047035
H04L 12/28		(43) 공개일자	2000년07월25일
(21) 출원번호	10-1998-0063784		
(22) 출원일자	1998년12월31일		

(71) 출원인 한국전기통신공사, 이계철
대한민국
463-010
경기도 성남시 분당구 정자동 206

(72) 발명자 설근석
대한민국
305-348
대전광역시 유성구 화암동 62-1

이해영
대한민국
305-348
대전광역시 유성구 화암동 62-1

정상현
대한민국
305-348
대전광역시 유성구 화암동 62-1

정학진
대한민국
305-348
대전광역시 유성구 화암동 62-1

(74) 대리인 이정훈
이후동

(77) 심사청구 있음

(54) 출원명 사용자 데이터 전송 및 수신을 위한 버퍼제어 방법

요약

본 발명은 ATM통신 시 사용자 데이터나 시그널링 데이터를 송/수신하는데 사용되는 AAL(ATM Adaptation Layer:적응층)타입5를 위해 윈도우즈 네트워크 커널내의 버퍼를 관리하는 적응층타입5사용자 데이터 전송 및 수신을 위한 버퍼제어 방법에 관한 것이다. 적응층타입5CPCS PDU 처리부는 응용프로그램 또는 상위레벨 디바이스 드라이버 (100) 와 소프트웨어 송신부 (200) 및 소프트웨어 수신부 (250) 를 사이에 두고 연결되는 ATM정합 보드를 구비한다. ATM정합 보드는 송신링 (320) 과 상태큐 (340) 를 갖는 보드 송신부 (300) 와, 프리버퍼 큐 (370) 및 상태큐 (390) 를 갖는 보드 수신부 (350) 를 구비하여, 소프트웨어 송신부 (220) 및 수신부 (270) 를 통하여 상위레벨 디바이스 드라이버 (100) 로 데이터 신호를 교환한다. 이에 따라 적응층타입5 CPCS PDU 처리부는 ATM 보드 내에서 적응층타입5를 지원하는 경우에 사용자 데이터를 효과적으로 송/수신할 수 있다. 또한 실시간 전송을 요하지 않는 여타의 적응층타입을 위해서도 효과적으로 사용될 수 있다.

대표도

도1

명세서

도면의 간단한 설명

도 1는 본 발명에 따른 소프트웨어 및 보드의 전체 구성을 설명하는 개략도,

도 2는 도 1의 처리부의 보드 송신부에 대한 자료구조를 나타낸 도면,

도 3는 보드 수신부의 자료구조를 나타낸 도면,

도 4는 본 발명에 따른 소프트웨어 송신부의 동작 수순을 나타낸 순서도로서,

도 4a는 소프트웨어 송신부 (200) 에서의 초기화,

도 4b는 소프트웨어 송신부 (200) 에서의 호/연결 설정 및 해제시,

도 4c는 소프트웨어 송신부 (200) 에서의 사용자로부터 데이터 송신 요구시 및, 도 4d는 소프트웨어 송신부 (200) 에서의 보드로부터 데이터 송신 완료시를 나타낸 도면.

도 5는 본 발명에 따른 소프트웨어 수신부의 동작 수순을 나타낸 순서도로서,

도 5a는 소프트웨어 수신부 (250) 에서의 수신부 초기화,

도 5b는 소프트웨어 수신부 (250) 에서의 호/연결 설정 및 해제시,

도 5c는 소프트웨어 수신부 (250) 에서의 보드로부터 데이터 수신 완료시,

및, 도 5d는 소프트웨어 수신부 (250) 사용자로부터 데이터 수신요구시를 나타낸 도면.

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

본 발명은 ATM(비동기 전송모드;Asynchronous Transfer Mode) 통신 시 사용자 데이터나 시그널링 데이터를 송/수신하는데 사용되는 ATM적응층타입 5의 사용자를 위해 윈도우즈 네트워크 커널내에서 데이터의 전송 및 수신하여 버퍼를 제어하는 방법에 관한 것이다.

두 ATM 단말간에 사용자 데이터를 주고 받으려면, AAL(ATM적응층;ATMAdaptation Layer) 계층을 통하여야 한다. 현재 AAL 계층은 타입(Type) 1 에서 타입(Type) 5까지 정의되어 있으며, 비실시간 데이터를 많은 오버헤드없이 전송하는 데에는 적응층타입 5가 많이 사용된다. 현재 적응층타입5데이터 전송을 위해서 적응층타입 5프로토콜을 지원하는 하드웨어를 사용하고 있다.

이러한 하드웨어의 도움을 받아서 데이터를 전송하기 위해서는 버퍼 관리가 필연적이며, 운영체제 커널내의 메모리를 적게 사용하면서 버퍼를 할당 및 해제하는 것이 중요하다.

또한 적응층타입5 사용자 데이터는 최대 64K 바이트 크기이며, 가변 길이를 갖는다. 이러한 특성을 갖는 적응층 5 사용자 데이터를 커널내의 자원을 최대한 줄이면서 송/수신하기 위해서는 송신 버퍼의 경우 데이터 전송 요구량에 따라 온-디멘드(On-demand) 방식으로 버퍼를 할당하고, 수신 버퍼의 경우 최소 버퍼량과 최대 버퍼량을 두어 버퍼를 할당하여야 한다.

발명이 이루고자 하는 기술적 과제

이에 따라 본 발명에서 사용하는 버퍼관리 기법을 사용하여, 실시간 전송을 요하지 않는 여타의 적응층타입을 위해 사용될 수 있도록, ATM 보드 내에서 적응층타입5를 지원하는 경우에 사용자 데이터를 효과적으로 송/수신할 수 있는 버퍼제어방법을 제공한 것을 제 1목적으로 한다.

또, 본 발명의 데이터의 전송 및 수신하여 버퍼를 제어하는 방법에 따라 데이터 송신의 경우 버퍼를 온-디멘드방식으로 할당함으로써 버퍼 사용량을 줄여 데이터의 송/수신을 제어할 수 있는 방법을 제공하는 것을 제 2목적으로 한다.

또한, 본 발명의 데이터의 버퍼 제어 방법에 따라, 데이터 수신의 경우 최대 버퍼 할당량을 둠으로써 버퍼 낭비를 막고, 최소 버퍼 할당량 및 최대 버퍼 할당량을 둠으로써 각 연결의 버퍼 필요량에 따라 적절히 할당 받아 데이터의 송/수신을 제어할 수 있는 방법을 제공하는 것을 제 3목적으로 한다.

데이터 송/수신의 경우 각 연결이 최소한의 버퍼를 할당 받을 수 있도록 보장하고, 하나의 연결이 버퍼를 독점적으로 사용하여 데이터의 송/수신을 제어할 수 있는 방법을 제공하는 것을 방지한 것을 제 3목적으로 한다.

발명의 구성 및 작용

본 발명은 ATM 통신 단말기의 B-ISDN 망 접속을 위한 망 접속 정합용 보드를 사용하여 사용자 데이터를 송/수신한다. 이때 비실시간 사용자 데이터 송/수신을 위하여 적응층타입5를 사용하며, 망 접속 정합용 보드는 적응층타입 5의 일부 기능을 제공한다.

도 1 은 적응층타입5를 사용하는 경우의 보드 및 소프트웨어의 전체 구조를 나타낸다.

도 1 에 있어서, 적응층타입5 CPCS PDU 처리부는 응용프로그램 또는 상위레벨 디바이스 드라이버 (100) 와 소프트웨어 송신부 (200) 및 소프트웨어 수신부 (250) 를 사이에 두고 연결되는 ATM정합 보드를 구비한다. ATM정합 보드는 송신링 (320) 과 상태큐 (340) 를 갖는 보드 송신부 (300) 와, 프리버퍼 큐 (370) 및 상태큐 (390) 를 갖는 보드 수신부 (350) 를 구비하여, 소프트웨어 송신부 (220) 및 수신부 (270) 를 통하여 상위레벨 디바이스 드라이버 (100) 로 데이터 신호를 교환한다.

또한, 적응층타입5CPCS PDU 처리부 각각의 기능은 다음과 같다.

보드 송신부 (300) 는 적응층타입5CPCS PDU 세그먼트(Segmentation)기능을 제공한다. 사용자 데이터인 CPCS PDU는 호스트 메모리 내에 위치하며, 소프트웨어 송신부 (200) 는 CPCS PDU 시작 주소 및 크기를 도 2에 표시된 "송신 링 (320) "의 한 엔트리에 써넣는다. 보드 송신부 (300) 는 "송신 링"의 엔트리를 자신의 메모리 영역 내로 복사하여, 이후에 소프트웨어 송신부 (200) 가 그 엔트리를 재사용할 수 있도록 한다. 소프트웨어 송신부 (200) 는 "송신 링"의 엔트리가 사용 가능한지를 먼저 확인하여야 하며, 사용 가능한 엔트리가 없는 경우에는 사용 가능한 엔트리가 생성될 때까지 데이터 송신을 멈추어야 한다. 보드 송신부 (300) 는 해당 사용자 데이터의 세그먼트가 끝나면 도 2의 "상태 큐 (30) "를 통하여 소프트웨어 송신부 (200) 에게 알려준다. 소프트웨어 송신부 (200) 는 해당 사용자 데이터를 담고 있는 버퍼를 해제할 수 있다. 도 2는 도 1의 처리부의 보드 송신부 (300) 에 대한 자료구조를 나타낸다.

소프트웨어 송신부 (200) 는 응용 프로그램이나 상위레벨 디바이스 드라이버로부터 데이터 전송을 요구 받는 경우 이를 다수의 CPCS PDU로 만든 다음 보드 송신부 (300) 의 "송신 링" 엔트리에 기록한다. 보드 송신부 (300) 로부터 전송 완료 신호를 받으면, 해당 버퍼를 해제한다. 소프트웨어 송신부 (200) 는 두 가지 자원, 즉 CPCS PDU를 생성하기 위한 버퍼와 "송신 링" 엔트리를 관리하여야 한다.

보드 수신부 (350) 는 적응층타입5CPCS PDU 리어셈블리(Reassembly)기능을 제공한다. 네트워크로부터 수신된 CPCS PDU를 저장하기 위하여 도 3의 프리버퍼큐("Free Buffer Queue")의 엔트리들이 가리키는 프리버퍼(Free Buffer)를 이용한다. 만약 CPCS PDU의 크기가 커서 둘 이상의 버퍼에 저장되어야 하는 경우 도 3에서와 같은 연결 리스트를 이용하여, 다수의 버퍼를 연결한다. CPCS PDU가 생성되면 보드 수신부 (350는 도 3에서의 "상태 큐 (30) "의 엔트리에 버퍼에 대한 정보를 저장한 후 인터럽트를 통하여 소프트웨어 수신부 (250) 에 전달한다. "상태 큐 (30) "의 엔트리는 연결 리스트 프리버퍼의 맨 처음 프리버퍼에 대한 포인터 정보, CPCS PDU의 크기 정보 및 해당 사용자를 구분하기 위한 VCCINDEX 값을 포함한다. 도3는 보드 수신부 (350) 의 자료구조를 나타낸다.

소프트웨어 수신부 (250) 는 중단없는 리어셈블리를 위해 항상 프리버퍼가 존재하도록 해야 한다. 프리버퍼는 호스트 메모리 내에 존재하며, 하나의 프리버퍼는 물리적으로 연속적이어야 한다. 소프트웨어 수신부 (250) 는 "프리버퍼 Queue"의 엔트리를 이용하여 보드 수신부 (350) 에 프리버퍼를 제공한다. 보드 수신부 (350) 로부터 인터럽트를 통해 새로운 CPCS PDU가 네트워크로부터 도착했다는 신호를 받으면, 소프트웨어 수신부 (250) 는 "상태 큐 (30) "의 엔트리를 조사하여 CPCS PDU의 시작주소 및 크기, 그리고 VCCINDEX 값을 호스트 메모리내로 복사하고 해당 "상태 큐 (30) " 엔트리를 보드 수신부 (350) 가 재사용할 수 있도록 한다. VCCINDEX 값을 이용하여 CPCS PDU를 수신할 응용 프로그램이나 상위레벨 디바이스 드라이버를 알아내고, 만약 수신을 대기 중이면 CPCS PDU를 넘겨주고, 대기 중이 아니라면 CPCS PDU를 해당 응용 프로그램이나 상위레벨 디바이스 드라이버의 수신 버퍼 큐 (30) 에 넣어둔다. 이후 해당 응용 프로그램이나 상위레벨 디바이스 드라이버가 수신을 요구하는 경우 수신 버퍼 큐 (30) 에 저장되어 있는 CPCS PDU를 사용자 영역의 버퍼로 복사한 후, CPCS PDU를 담고 있는 버퍼를 해제한다.

소프트웨어 송신부 (200) 는 사용자의 데이터 전송 요구에 따라 버퍼를 할당하며, 다음과 같은 처리수순에 따른다. "송신 링"의 엔트리 개수 * CPCS PDU의 최대크기를 전체 버퍼의 상한으로 잡는다. 현재 연결 설정이 완료된 적응층타입5연결의 개수를 N이라 하면, (전체 버퍼 상한의 50%) / N을 각 연결의 버퍼 하한으로 잡는다. 이제 각 연결에는 위 하한만큼의 버퍼를 할당할 수 있으며, 만약 하한을 초과하여 할당을 요청하는 경우에는 (전체 버퍼 상한의 나머지 50%) 내에서 계속 추가 할당하도록 한다. 만약 기존의 연결 설정이 해제되는 경우에는 각 연결의 하한을 올려주면 된다. 새로운 연결 설정이 완료된 경우에는 기존 연결의 버퍼 하한을 낮추어서 버퍼 할당량을 받아온다. 이 경우 버퍼 할당량이 전체 버퍼 상한을 초과하는 경우가 발생할 수 있으나, 일시적인 현상이 될 것이므로 크게 문제되지 않는다.

소프트웨어 송신부 (200) 의 구체적인 처리수순은 아래와 같이 도4를 참고한다.

우선, 소프트웨어 송신부 (200) 의 상수 및 변수를 설정한다.

상수 : NumOfBuffer -> 전체 버퍼의 상한 개수, 즉 "송신 링" 엔트리의 개수

변수 : NumOfConnection -> 전체 연결의 개수

CurrentLowerLimit -> 현재 각 연결의 버퍼 하한,

즉 ? (NumOfBuffer/2)/NumOfConnection?

HalfNumOfBuffer -> 전체 버퍼 상한의 나머지 50%의 개수

즉, NumOfBuffer - CurrentLowerLimit * NumOfConnection

FreeNumOfBufer -> 전체 버퍼 상한의 나머지 50%의 개수 중 현재 사용 가능한 버퍼 개수

이어서, 도 4에 있어서, 소프트웨어 송신부 (２００) 의 각 자료의 연결구조를 아래와 같이 정한다.

각 연결의 자료구조 :

CurrentNumOfAssignedBuffer -> 현재 할당된 버퍼 수

NumOfBufferOverLowerLimit -> CurrentLowerLimit를 초과하여 할당된 버퍼 수

즉, CurrentNumOfAssignedBuffer - CurrentLowerLimit

VCCINDEX : 연결의 VCC identifier 값

도 4-1에 있어서, 소프트웨어 송신부 (２００) : 초기화를 설정한다.(스탭410) NumOfBuffer = 송신링 엔트리의 개수;

NumOfConnection = 0;

CurrentLowerLimit = NumOfBuffer /2;

HalfNumOfBuffer = NumOfBuffer /2;

FreeNumOfBuffer = HalfNumOfBuffer;

이와 같이 초기화를 설정한후, 도 4-2 소프트웨어 송신부 (２００) : 호/연결 설정및 해제시에 그 처리 수순이 다음과 같이 진행한다.

입력: 새로운 연결 또는 기존 연결의 VCCINDEX(VCC indentifier)(스탭420)

처리사항이 연결설정에 관한 것인지 아니면 해제에 관한 것인지 결정한다. (스탭440)

스텝 440에서 연결설정이 완료한 경우라면,

NumOfConnection = NumOfConnection + 1;

OldLowerLimit = CurrentLowerLimit;

CurrentLowerLimit = (NumOfBuffer/2)/NumOfConnection?;

NewNum = NumOfBuffer - CurrenLowerLimit * NumOfConnection;

DiffNum = NewNum - HalfNumOfBuffer;

HalfNumOfBuffer = NewNum;

(각 연결의 NumOfBufferOverLowerLimit) = (각 연결의 NumOfBufferOverLowerLimit) + (OldLowerLimit - CurrentLowerLimit);

FreeNumOfBuffer = FreeNumOfBuffer + DiffNum - (각 연결의 NumOfBufferOverLowerLimit의 증가분);

와 같이 처리한다. (스탭443)

그후 (스탭443)에 이어서 새로운 연결 자료구조를 만드는데, 새로운 연결 자료구조의 내용을 다음과 같이 채운다.

CurrentNumOfAssignedBuffer = 0;

NumOfBufferOverLowerLimit = 0;

VCCINDEX = value of VCC identifier; (스탭445) 이와 같이 자료구조를 형성하면, (스탭445)의 처리 수순은 종료한다.

한편, 이와 같이 스텝 410에서 초기화를 설정한후, 소프트웨어 송신부 (２００) : 호/연결 해제시에 그 처리 수순이 다음과 같이 진행한다.

스텝 440에서 연결 설정이 해제되는 경우라면,

NumOfConnection = NumOfConnection - 1; (스탭442)

스텝 442의 연산이 성립한다면, 해당 연결의 자료구조를 찾고, 현재 대기중인 송신 버퍼를 모두 해제하고, 연결 자료구조를 해제한다. 즉,

FreeNumOfBuffer = FreeNumOfBuffer +(연결의NumOfBufferOverLowerLimit); (스탭444)

그후, NumOfConnection == 0? 를 판단한다. (스탭446)

NumOfConnection == 0? 가 그렇다고 판단한다면, (스탭448)은,

CurrentLowerLimit = NumOfBuffer/2;

HalfNumOfBuffer = NumOfBuffer /2;

FreeNumOfBuffer = HalfNumOfBuffer;

(스탭448) 을 통하여 그 처리가 종료한다.

그러나 스텝 446에서 NumOfConnection == 0? 를 그렇지 않다고 판단한다면, (스탭449)는,

CurrentLowerLimit = (NumOfBuffer/2)/NumOfConnection?;

NewNum = NumOfBuffer - CurrenLowerLimit * NumOfConnection;

DiffNum = NewNum - HalfNumOfBuffer;

HalfNumOfBuffer = NewNum;

(각 연결의 NumOfBufferOverLowerLimit) =(각 연결의 NumOfBufferOverLowerLimit) +

(OldLowerLimit - CurrentLowerLimit);

FreeNumOfBuffer = FreeNumOfBuffer + DiffNum + (각 연결의 NumOfBufferOverLowerLimit의 감소분);

(스텝449)을 통하여 도 4-2a 소프트웨어 송신부 (２００에 대한 호/설정 및 해제처리가 종료한다.

도 4b에 의해 호/연결 설정이 완료한 후에, 도 4-3 소프트웨어 송신부 (２００) : 사용자로부터 데이터 송신 요구시의 처리는 다음과 같다.

입력으로서, 송신요구 버퍼의 개수(NumOfInput)를 판단한다. (스텝460)

이후 VCCINDEX 값에 의해 해당 연결의 자료구조를 찾는다.(스텝462)

그 후, FreeNumBuffer <= 0 and (연결의 CurrentNumOfAssignedBuffer) >= CurrentLowerLimit 여부를 판단한다. (스텝 464)

스텝 464에서 그 판단이 예 이라고 하면, 데이터의 송신 요구 거절하고 종료한다.(스텝 466)

한편, 스텝 464에서 그 판단이 아니오 이라고 하면, (연결의 CurrentNumOfAssignedBuffer) < CurrentLowerLimit 을 판단한다. (스텝 470)

스텝 470에서 그 판단이 아니오 이라고 하면, FreeNumOfBuffer = FreeNumOfBuffer - NumOfInput ;

(연결의 NumOfBufferOverLowerLimit) = (연결의 NumOfBufferOverLowerLimit) +

NumOfInput; (스텝 471) 으로 처리되고 그 제어가 (스텝 472)로 넘어간다.

스텝 470에서 그 판단이 예 이라고 하면, (연결의 CurrentNumOfAssignedBuffer) = (연결의 CurrentNumOfAssignedBuffer + NumOfInput; (스텝 472)

그 후, 스텝 474에서는 버퍼를 NumOfInput만큼할당한다면, 보드송신부(300)의 송신링의 엔트리에 버퍼에 대한 정보를 기록한후, 보드 송신부(300)에게 데이터 송신 요구를 한다. 스텝 474의 처리가 종료한다.

도 4b에 의해 호/연결 설정이 완료한 후에, 도 4-4 소프트웨어 , 송신부 (200) : 보드로부터 데이터 송신 완료시의 처리는 다음과 같다.

먼저, 보드 송신부(300)의 상태큐 엔트리를 호스트 메모리로 복사한후, 해당 상태큐 엔트리를 보드 송신부(300)가 재사용할 수 있도록 한다. (스텝480)

또한, 상태큐 엔트리의 VCCINDEX 정보로부터 해당 연결의 자료구조를 찾는다. 이때NumOfInput = (송신 완료된 버퍼의 개수) 으로 처리한다.(스텝481)

이 후, (연결의 NumOfBufferOverLowerLimit) > 0여부를 판단한다. (스텝 483)

스텝 483에서 그 판단이 예 이라고 하면, FreeNumOfBuffer = FreeNumOfBuffer + NumOfInput ;

(연결의 NumOfBufferOverLowerLimit) = (연결의 NumOfBufferOverLowerLimit) - NumOfInput; 으로 처리하고 그 제어가 (스탭 487)로 넘어간다.

한편, 스텝 483에서 그 판단이 아니오 이라고 하면, (스탭 487)에서 (연결의 CurrentNumOfAssignedBuffer) = (연결의 CurrentNumOfAssignedBuffer) - NumOfInput; 으로 처리한다.(스텝 488)

그후, 스텝 489에서는 송신 완료된 해당 버퍼를 해제하고, 스텝 489의 처리가 종료한다.

소프트웨어 수신부 (250) 는 송신부의 경우와 달리 미리 각 적응층타입5연결을 위해 버퍼를 할당하여야 한다. 따라서 전체 버퍼 상한만큼의 버퍼를 미리 할당한다. 전체 버퍼 상한은 "프리버퍼 Queue"엔트리의 개수 * 한 페이지의 크기로 잡는다. 현재 연결 설정이 완료된 적응층타입5연결의 개수를 N이라 하면, (전체 버퍼 상한의 50%) / N을 각 연결의 버퍼 하한으로 잡는다. 이제 각 연결에는 위 하한만큼의 버퍼를 할당할 수 있으며, 만약 하한을 초과하여 할당을 요청하는 경우에는 (전체 버퍼 상한) / N 만큼을 (전체 버퍼 상한의 나머지 50%)에서 계속해서 할당 받도록 한다. 이제 각 연결은 버퍼 하한만큼을 확보할 수 있게 되었으며, 각 연결의 버퍼 상한은 버퍼 하한의 3배가 된다. 만약 기존의 연결 설정이 해제되는 경우에는 각 연결의 하한을 올려주면 된다. 그러나, 새로운 연결 설정이 완료된 경우에는 기존 연결의 버퍼 하한을 낮추며, 동시에 버퍼 상한을 초과하여 할당받은 연결에 대해서는 버퍼를 해제하도록 한다. 해제 순서는 먼저 할당받은 버퍼를 해제하도록 한다. 소프트웨어 수신부 (250) 의 구체적인 처리수순은 아래와 같이 도 5를 참고한다.

우선, 도 5에 있어서, 소프트웨어 수신부 (250) 의 상수 및 변수를 설정한다.

상수 : NumOfBuffer -> 전체 버퍼의 상한 개수, 즉 "프리버퍼 Queue" 엔트리의 개수

변수 : NumOfConnection -> 전체 연결의 개수

CurrentLowerLimit -> 현재 각 연결의 버퍼 하한,

즉 ?(NumOfBuffer/2)/NumOfConnection?

CurrentUpperLimit -> 현재 각 연결의 버퍼 상한

즉 CurrentLowerLimit * 3, 만약 이 값이 NumOfBuffer보다 크면 NumOfBuffer

HalfNumOfBuffer -> 전체 버퍼 상한의 나머지 50%의 개수

즉, NumOfBuffer - CurrentLowerLimit * NumOfConnection

FreeNumOfBufer -> 전체 버퍼 상한의 나머지 50%의 개수 중 현재 사용 가능한 버퍼 개수

이어서, 도 5의 소프트웨어 수신부 (250) 의 각 자료의 연결구조를 아래와 같이 정한다.

각 연결의 자료구조 :

CurrentNumOfPendingBuffer -> 현재 사용자가 수신하기를 기다리는 버퍼 수

NumOfBufferOverLowerLimit -> CurrentLowerLimit를 초과하여 할당된 버퍼 수

즉, CurrentNumOfAssignedBuffer - CurrentLowerLimit

VCCINDEX : 연결의 VCC identifier 값

도 5-1에 있어서, 소프트웨어 수신부 (250) : 초기화를 설정한다.(스탭510) NumOfBuffer = "프리버퍼 Queue"엔트리의 개수;

NumOfConnection = 0;

CurrentLowerLimit = NumOfBuffer /2;

CurrentUpperLimit = NumOfBuffer;

HalfNumOfBuffer = NumOfBuffer /2;

FreeNumOfBuffer = HalfNumOfBuffer

이와 같이 초기화를 설정한후, 도5-2 소프트웨어 수신부 (250) : 호/연결 설정및 해제시에 그 처리 수순이 다음과 같이 진행한다.

입력: 새로운 연결 또는 기존 연결의 VCCINDEX(VCC indentifier)(스탭520)

처리사항이 연결설정에 관한 것인지 아니면 해제에 관한 것인지 결정한다. (스탭522)

스텝 522에서 연결설정이 완료한 경우라면,

NumOfConnection = NumOfConnection + 1;

OldLowerLimit = CurrentLowerLimit;

CurrentLowerLimit = ?(NumOfBuffer/2)/NumOfConnection?;

CurrentUpperLimit = NewNum * 3;

if (CurrentUpperLimit > NumOfBuffer)

CurrentUpperLimit = NumOfBuffer;

NewNum = NumOfBuffer - CurrenLowerLimit * NumOfConnection;

DiffNum = NewNum - HalfNumOfBuffer;

HalfNumOfBuffer = NewNum;

(각 연결의 NumOfBufferOverLowerLimit) = (각 연결의 NumOfBufferOverLowerLimit) + (OldLowerLimit - CurrentLowerLimit);

if ((각 연결의 CurrentNumOfPendingBuffer) > CurrentUpperLimit) {

CurrentNumOfPendingBuffer = CurrentUpperLimit);

차이만큼의 해당 버퍼를 해제하고, 보드 수신부(350)의 "프리버퍼 큐(30)에 되돌려 준다;

(각 연결의NumOfBufferOverLowerLimit) 를 위 차이만큼 감소 시킨다

}

FreeNumOfBuffer = FreeNumOfBuffer + DiffNum - (각 연결의 NumOfBufferOverLowerLimit의 증가분);

와 같이 처리한다. (스탭524)

그후 (스탭524)에 이어서 새로운 연결 자료구조를 만드는데, 새로운 연결 자료구조의 내용을 다음과 같이 채운다.

CurrentNumOfPendingBuffer = 0;

NumOfBufferOverLowerLimit = 0;

VCCINDEX = value of VCC identifier; (스탭534)

이와 같이 자료구조를 형성하면, (스탭534)의 처리 수순은 종료한다.

한편, 이와 같이 스텝 510에서 초기화를 설정한후, 소프트웨어 송신부 (200) : 호/연결 해제시에 그 처리 수순이 다음과 같이 진행한다.

스텝 522에서 연결 설정이 해제되는 경우라면,

NumOfConnection = NumOfConnection - 1; (스탭526)

스텝 526의 연산이 성립한다면, 해당 연결의 자료구조를 찾고, 현재 대기중인 수신 버퍼를 모두 해제하고, 연결 자료구조를 해제한다. 그리고,

FreeNumOfBuffer = FreeNumOfBuffer +(연결의NumOfBufferOverLowerLimit);와 같이 처리한다 (스탭527)

그후, NumOfConnection == 0? 를 판단한다. (스탭528)

(스탭528)에서 NumOfConnection == 0? 가 그렇다고 판단한다면, (스탭530)은,

CurrentLowerLimit = NumOfBuffer/2;

HalfNumOfBuffer = NumOfBuffer /2;

CurrenUpperLimit = NumOfBuffer;

FreeNumOfBuffer = HalfNumOfBuffer;

(스탭530) 을 통하여 그 처리가 종료한다.

그러나 (스텝 528)에서 NumOfConnection == 0? 를 그렇지 않다고 판단한다면, (스탭532)는,

CurrentLowerLimit =

?(NumOfBuffer/2)/NumOfConnection?;

CurrentUpperLimit = NewNum * 3;

if (CurrentUpperLimit > NumOfBuffer)

CurrentUpperLimit = NumOfBuffer;

NewNum = NumOfBuffer - CurrenLowerLimit * NumOfConnection;

DiffNum = NewNum - HalfNumOfBuffer;

HalfNumOfBuffer = NewNum;

(각 연결의 NumOfBufferOverLowerLimit) = (각 연결의 NumOfBufferOverLowerLimit) + (OldLowerLimit - CurrentLowerLimit);

FreeNumOfBuffer = FreeNumOfBuffer + DiffNum + (각 연결의 NumOfBufferOverLowerLimit의 감소분);

(스탭532)을 통하여 도 5b 소프트웨어 수신부 (250) 에 대한 호/설정 및 해제처리가 종료한다.

도 5b에 의해 호/연결 설정이 완료한 후에, 도 5c 소프트웨어 수신부 (250) : 보드로부터 데이터 수신 완료시의 처리는 다음과 같다.

먼저 보드 수신부(350)의 상태큐 엔트리를 호스트 메모리로 복사한후, 해당 상태큐 엔트리를 보드 수신부(350)가 재사용할 수 있도록 한다. (스텝540)

이후 상태큐 엔트리의 VCCINDEX 정보로부터 해당 연결의 자료구조를 찾고, NumOfInput = (수신 요구된 버퍼의 개수); 와 같이 처리한다.(스탭 542)

이제, (연결의 CurrentNumOfPendingBuffer) == CurrentUpperLimit 여부를 판단한다. (스텝 544)

스텝 544에서 그 판단이 예 이라고 하면, 해당 연결의 수신 대기 큐에서 NumOfInput만큼을 해제하고, 해제한 버퍼는 보드 수신부(350)의 "프리버퍼 Queue(30)"에 되돌려 준다.(스탭 548)

스탭 548의 처리가 끝나면 그 제어가 (스탭570)으로 넘어간다.

한편, 스텝 544에서 그 판단이 아니오 이라고 하면, (연결의 CurrentNumOfPendingBuffer) == 0 and 해당 사용자가 수신대기중 인지를 판단한다. (스텝 550)

스텝 550에서 그 판단이 예 이라고 하면, 사용자의 수신 요구를 만족시켜 준다. 그리고, NumOfInput만큼을 해제하고, 해제한 버퍼는 보드 수신부(350)의 "프리버퍼 Queue(30)"에 되돌려 준후 그 처리가 종료한다.(스탭 552)

스텝 550에서 그 판단이 아니오 이라고 하면, (연결의 CurrentNumOfPendingBuffer) < CurrentLowerLimit인지를 판단한다. (스텝 560)

스탭 560에서 그 판단이 아니오 이라고 하면, (스탭 562)에서

(해당 연결의 CurrentNumOfPendingBuffer) = (해당 연결의 CurrentNumOfPendingBuffer) + NumOfInput;

(해당 연결의 NumOfBufferOverLoweLimit) = (해당 연결의 NumOfBufferOverLoweLimit) + NumOfInput;

FreeNumOfBuffer = FreeNumOfBuffer - NumOfInput;

와 같이 처리하고 그 제어가 (스텝 570)으로 넘어간다.

스탭 560에서 그 판단이 예 이라고 하면, (스탭 564)에서

(해당 연결의 CurrentNumOfPendingBuffer) = (해당 연결의 CurrentNumOfPendingBuffer) + NumOfInput;

와 같이 처리하고 그 제어가 (스텝 570)으로 넘어간다.

(스텝 570)에서는 새로운 수신 버퍼(NumOfInput 만큼의 개수)를 해당 연결의 수신 대기 큐에 집어넣는다. 스텝 570의 처리가 종료한다.

도 5b에 의해 호/연결 설정이 완료한 후에, 도 5-4 소프트웨어 수신부 (250) : 사용자로부터 데이터 수신 요구시의 처리는 다음과 같다.

먼저, VCCINDEX 정보로부터 해당 연결의 자료구조를 찾는다. (스텝580)

이 후, (연결의 CurrentNumOfPendingBuffer) == 0여부를 판단한다. (스텝 582)

스텝 582에서 그 판단이 예 이라고 하면, 사용자의 수신 요구를 대기시키고 그 처리를 종료한다.(스탭 584)

한편, 스텝 582에서 그 판단이 아니오 이라고 하면, (스탭 586)에서 사용자의 수신 요구를 만족시켜 준 후, 수신이 끝난 버퍼(개수: NumOfInput)는 해제하고, 해제한 버퍼는 보드 수신부(350)의 "프리버퍼 Queue(30)"에 되돌려 준다.(스탭 586)

스탭 586의 처리가 끝나면, 다시 (연결의 NumOfBufferOverLowerLimit) > 0인지를 판단한다.(스탭 590)

스텝 590에서 그 판단이 예 이라고 하면, (스탭 592)에서

FreeNumOfBuffer = FreeNumOfBuffer + NumOfInput ;

(연결의 NumOfBufferOverLowerLimit) = (연결의 NumOfBufferOverLowerLimit) - NumOfInput;

와 같이 처리하고, 그 제어가 스탭 595로 넘어간다.

한편, 스텝 590에서 그 판단이 아니오 이라고 하면, (스탭 595)에서

(연결의 CurrentNumOfPendingBuffer) = (연결의 CurrentNumOfPendingBuffer) - NumOfInput;.

와 같이 처리하고, 그 처리를 종료한다.

발명의 효과

본 발명에서 사용하는 버퍼관리 기법을 사용하면 다음과 같은 효과가 있다.

첫째, 데이터 송신의 경우 버퍼를 On-Demand 방식으로 할당함으로써 버퍼 사용량을 줄인다.

둘째, 데이터 수신의 경우 최대 버퍼 할당량을 둠으로써 버퍼 낭비를 막고, 최소 버퍼 할당량 및 최대 버퍼 할당량을 둠으로써 각 연결의 버퍼 필요량에 따라 적절히 할당 받는다.

마지막으로, 데이터 송/수신의 경우 각 연결이 최소한의 버퍼를 할당 받을 수 있도록 보장하고, 하나의 연결이 버퍼를 독점적으로 사용하는 것을 방지한다.

(57) 청구의 범위

청구항 1.

적응층타입5CPCS PDU 처리부를 갖는 버퍼제어 방법에 있어서,

응용프로그램 또는 상위레벨 디바이스 드라이버 (100) 와;

및 소프트웨어 송신부 (220) 및 소프트웨어 수신부 (250) 를 사이에 두고 연결되는 ATM정합 보드를 구비한 소프트웨어 송신부 (200)와;

ATM정합 보드는 송신링 (320) 과 상태큐 (340) 를 갖는 보드 송신부 (300) 와;

프리버퍼 큐 (370) 및 상태큐 (390) 를 갖는 보드 수신부 (350) 를 구비하고,

상기 소프트웨어 송신부 (220) 및 수신부 (270) 를 통하여 상위레벨 디바이스 드라이버 (100) 로 데이터 신호를 교환 할 때, 상기 송신부 (200) 는 CPCS PDU 시작 주소 및 크기를 "송신 링 (320) "의 한 엔트리에 써넣는 단계;

상기 써넣는 단계다음에, 상기 보드 송신부 (300) 는 "송신 링"의 엔트리를 자신의 메모리 영역 내로 복사하여, 이후에 소프트웨어 송신부 (200) 가 그 엔트리를 재사용하는 단계;

상기 재사용단계로부터, "송신 링"의 엔트리가 사용 가능한지를 먼저 확인하여, 사용 가능한 엔트리가 없는 경우에는 사용 가능한 엔트리가 생성될 때까지 데이터 송신을 멈추는 단계;

그후, 해당 사용자 데이터의 세그먼트가 끝나면 도 2의 "상태 큐 (30) "를 통하여 소프트웨어 송신부 (200) 에게 알려주는 단계를 포함하는 것을 특징으로 하는 버퍼 제어 방법.

청구항 2.

청구항 1에 있어서,

상기 소프트웨어 송신부 (200) 는 해당 사용자 데이터를 담고 있는 버퍼를 해제하고, 응용 프로그램이나 상위레벨 디바이스 드라이버로부터 데이터 전송을 요구 받는 경우 이를 다수의 CPCS PDU로 만든 다음 보드 송신부 (300) 의 "송신 링" 엔트리에 기록하는 단계를 포함하는 것을 특징으로 하는 버퍼 제어 방법.

청구항 3.

청구항 1에 있어서,

상기 보드 수신부 (350) 는 네트워크로부터 수신된 CPCS PDU를 저장하기 위하여 도 3의 "프리버퍼 큐"의 엔트리들이 가리키는 프리버퍼를 이용하고, CPCS PDU가 생성되면 "상태 큐 (30) "의 엔트리에 버퍼에 대한 정보를 저장한 후 인터럽트를 통하여 소프트웨어 수신부 (250) 에 전달하는 단계를 포함하는 것을 특징으로 하는 버퍼 제어 방법.

청구항 4.

청구항 1에 있어서,

상기 "상태 큐 (30) "의 엔트리는 연결 리스트 프리버퍼의 맨 처음 프리버퍼에 대한 포인터 정보, CPCS PDU의 크기 정보 및 해당 사용자를 구분하기 위한 VCCINDEX 값을 포함하는 것을 특징으로 하는 버퍼 제어 방법.

청구항 5.

청구항 1에 있어서,

상기 만약 CPCS PDU의 크기가 커서 둘 이상의 버퍼에 저장되어야 하는 경우 연결 리스트를 이용하여, 다수의 버퍼를 연결하는 단계를 포함하는 것을 특징으로 하는 버퍼 제어 방법.

청구항 6.

청구항 1에 있어서,

상기 소프트웨어 수신부 (250) 는 중단없는 리어셈블리를 위해 항상 호스트 메모리 내에서 프리버퍼가 존재하도록 하는 단계를 포함하는 것을 특징으로 하는 버퍼 제어 방법.

청구항 7.

청구항 1에 있어서,

상기 소프트웨어 수신부 (250) 는 "프리버퍼 큐"의 엔트리를 이용하여 보드 수신부 (350) 에 프리버퍼를 제공하고, 보드 수신부 (350) 로부터 인터럽트를 통해 새로운 CPCS PDU가 네트워크로부터 도착했다는 신호를 받으면, 소프트웨어 수신부 (250) 는 "상태 큐 (30) "의 엔트리를 조사하여 CPCS PDU의 시작주소 및 크기, 그리고 VCCINDEX 값을 호스트 메모리내로 복사하고 해당 "상태 큐 (30) " 엔트리를 보드 수신부 (350) 가 재사용할 수 있도록 하는 단계를 포함하는 것을 특징으로 하는 버퍼 제어 방법.

청구항 8.
청구항 1에 있어서,

상기 VCCINDEX 값을 이용하여 CPCS PDU를 수신할 응용 프로그램이나 상위레벨 디바이스 드라이버를 알아내고, 만약 수신을 대기 중이면 CPCS PDU를 넘겨주고, 대기 중이 아니라면 CPCS PDU를 해당 응용 프로그램이나 상위레벨 디바이스 드라이버의 수신 버퍼 큐 (30) 에 넣어둔다. 이후 해당 응용 프로그램이나 상위레벨 디바이스 드라이버가 수신을 요구하는 경우 수신 버퍼 큐 (30) 에 저장되어 있는 CPCS PDU를 사용자 영역의 버퍼로 복사한 후, CPCS PDU를 담고 있는 버퍼를 해제하는 단계를 포함하는 것을 특징으로 하는 버퍼 제어 방법.

청구항 9.
청구항 1에 있어서,

상기 소프트웨어 송신부 (200) 는 사용자의 데이터 전송 요구에 따라 버퍼를 할당하는데,

"송신 링"의 엔트리 개수 * CPCS PDU의 최대크기를 전체 버퍼의 상한으로 잡는 단계;

현재 연결 설정이 완료된 적응층타입5연결의 개수를 N이라 하면, (전체 버퍼 상한의 50%) / N을 각 연결의 버퍼 하한으로 잡는 단계;

이제 각 연결에는 위 하한만큼의 버퍼를 할당할 수 있으며, 만약 하한을 초과하여 할당을 요청하는 경우에는 (전체 버퍼 상한의 나머지 50%)내에서 계속 추가 할당하는 단계;

만약 기존의 연결 설정이 해제되는 경우에는 각 연결의 하한을 올려주는 단계;

새로운 연결 설정이 완료된 경우에는 기존 연결의 버퍼 하한을 낮추어서 버퍼 할당량을 받아오는 단계를 포함하는 것을 특징으로 하는 버퍼 제어 방법.

청구항 10.
청구항 9에 있어서,

상기 소프트웨어 송신부(200)는 사용자로부터 데이터 송신 요구가 있는 경우, 송신 요구 버퍼 개수를 입력으로 하여 해당 연결의 자료구조를 찾는 제 1단계와;

전체 버퍼 상한의 나머지 50%의 개수 중 할당하지 않은 버퍼가 존재하는지 그리고, 해당 연결에 버퍼 하한을 초과하여 할당한 버퍼의 개수가 각 연결의 버퍼 하한 개수를 초과하는 지를 판별하는 제 2단계와;

상기 제 2단계에서 두 가지 모두 참인 경우에는 데이터 송신 요구를 거절하고 종료하는 제 3단계와;

상기 제 2단계에서 그렇지 않은 경우에는 이제 해당 연결에 버퍼 하한을 초과하여 할당한 버퍼의 개수가 각 연결의 버퍼 하한 개수를 초과하는 지를 판별하는 제 4단계와;

상기 제 4단계에서 거짓인 경우에는 전체 버퍼 상한의 나머지 50%의 개수 중 할당하지 않은 버퍼의 개수를 송신 요구 버퍼 개수 만큼 줄이고, 해당 연결의 버퍼 하한을 초과하여 할당한 버퍼의 개수를 송신 요구 버퍼 개수 만큼 늘이는 제 5단계와;

상기 제 4단계에서 참인 경우와 제 5단계를 마친 경우에는 해당 연결의 현재 할당된 버퍼수를 송신 요구 버퍼 개수만큼 늘이는 제 6단계와;

버퍼를 송신 요구 버퍼 개수만큼 할당하여, 보드송신부(300)의 송신링의 엔트리에 버퍼에 대한 정보를 기록한후, 보드 송신부(300)에게 데이터 송신 요구를 하는 제 7단계를 포함하는 것을 특징으로 하는 버퍼제어 방법

청구항 11.
청구항 9에 있어서,

상기 소프트웨어 송신부(200)는 보드부터 데이터 송신 완료신호가 있는 경우, 보드 송신부(300)의 상태큐 엔트리를 호스트 메모리로 복사한후, 해당 상태큐 엔트리를 보드 송신부(300)가 재사용할 수 있도록 하는 제 1단계와;

상태큐 엔트리의 VCCINDEX 정보로부터 해당 연결의 자료구조를 찾고, 송신 완료된 버퍼의 개수 정보를 알아내는 제 2 단계와;

해당 연결에 버퍼 하한을 초과하여 할당한 버퍼가 존재하는지 판별하는 제 3단계와;

상기 제 3단계에서 참인 경우에는 전체 버퍼 상한의 나머지 50%의 개수 중 할당하지 않은 버퍼의 개수를 송신 완료 버퍼 개수 만큼 늘이고, 해당 연결의 버퍼 하한을 초과하여 할당한 버퍼의 개수를 송신 완료 버퍼 개수 만큼 줄이는 제 4단계와;

상기 제 3단계에서 거짓인 경우와 상기 제4단계의 수행이 끝난 경우에는 해당 연결의 현재 할당된 버퍼수를 송신 완료 버퍼 개수만큼 줄이는 제 5단계와;

송신 완료된 해당 버퍼를 해제하는 제 6단계를 포함하는 것을 특징으로 하는 버퍼제어 방법

청구항 12.
청구항 1에 있어서,

상기 소프트웨어 수신부 (250) 는 "프리버퍼 큐"엔트리의 개수 * 한 페이지의 크기를 전체 버퍼 상한으로 잡는 단계;

현재 연결 설정이 완료된 적응층타입5연결의 개수를 N이라 하면, (전체 버퍼 상한의 50%) / N을 각 연결의 버퍼 하한으로 잡는 단계;

이제 각 연결에는 위 하한만큼의 버퍼를 할당할 수 있으며, 만약 하한을 초과하여 할당을 요청하는 경우에는 (전체 버퍼 상한) / N 만큼을 (전체 버퍼 상한의 나머지 50%)에서 계속해서 할당 받도록 하는 단계;

이제 각 연결은 버퍼 하한만큼을 확보할 수 있게 되었으며, 각 연결의 버퍼 상한은 버퍼 하한의 3배로 잡는 단계;

만약 기존의 연결 설정이 해제되는 경우에는 각 연결의 하한을 올려주는 단계;

그러나, 새로운 연결 설정이 완료된 경우에는 기존 연결의 버퍼 하한을 낮추며, 동시에 버퍼 상한을 초과하여 할당받은 연결에 대해서는 버퍼를 해제하도록 하는 단계;

이 경우 해제 순서는 먼저 할당받은 버퍼를 해제하도록 하는 단계를 포함하는 것을 특징으로 하는 버퍼 제어 방법

청구항 13.
청구항 12에 있어서,

상기 소프트웨어 수신부(250)는 보드로부터 데이터 수신 완료 신호가 있는 경우, 보드 수신부(350)의 상태큐 엔트리를 호스트 메모리로 복사한후, 해당 상태큐 엔트리를 보드 수신부(350)가 재사용할 수 있도록 하는 제 1단계와;

상태큐 엔트리의 VCCINDEX 정보로부터 해당 연결의 자료구조를 찾고, 수신 요구된 버퍼의 개수 정보를 알아내는 제 2 단계와;

해당 연결의 사용자가 수신하기를 기다리는 버퍼의 개수가 현재 각 연결의 버퍼 상한과 같은 지를 판별하는 제 3단계와;

상기 제 3단계에서 참인 경우에는 해당 연결의 수신 대기 큐에서 NumOfInput만큼을 해제하고, 해제한 버퍼는 보드 수신부(350)의 "프리버퍼 큐(30)"에 되돌려 주는 제 4단계와;

상기 제 3단계에서 거짓인 경우에는 해당 연결의 사용자가 수신하기를 기다리는 버퍼의 개수가 '0'인지 그리고, 해당 사용자가 수신 대기중인지를 판별하는 제 5단계와;

상기 제 5단계에서 참인 경우에는 사용자의 수신 요구를 만족시켜 주고, 수신 요구된 버퍼의 개수만큼의 해당 버퍼를 해제하고, 해제한 버퍼는 보드 수신부(350)의 "프리버퍼 큐(30)"에 되돌려 주고 종료하는 제 6단계와;

상기 제 5단계에서 거짓인 경우에는 해당 연결의 사용자가 수신하기를 기다리는 버퍼의 개수가 현재 각 연결의 버퍼 하한보다 작은지를 판별하는 제 7단계와;

상기 제 7단계에서 거짓인 경우에는 해당 연결의 사용자가 수신하기를 기다리는 버퍼의 개수 및 해당 연결의 버퍼하한을 초과하여 할당한 버퍼의 개수를 수신 요구된 버퍼의 개수만큼 각각 늘이고, 전체 버퍼 상한의 개수중 나머지 50%의 개수를 수신 요구된 버퍼의 개수만큼 줄이는 제 8단계와;

상기 제 7단계에서 참인 경우에는 해당 연결의 사용자가 수신하기를 기다리는 버퍼의 개수를 수신 요구된 버퍼의 개수만큼 각각 늘이는 제 9단계와;

상기 제4단계, 제8단계 및 제 9단계의 수행이 끝난 경우에는 수신 요구된 버퍼의 개수 만큼의 새로운 수신 버퍼를 해당 연결의 수신 대기 큐에 집어넣는 제 10단계를 포함하는 것을 특징으로 하는 버퍼제어 방법

청구항 14.

청구항 12에 있어서,

상기 소프트웨어 수신부(250)는 사용자로부터 데이터 수신 요구가 있는 경우, 해당 연결의 자료구조를 찾는 제 1단계와;

해당 연결의 사용자가 수신하기를 기다리는 버퍼의 개수가 '0'인지를 판별하는 제 2단계와;

상기 제 2단계에서 참인 경우에는 사용자의 수신 요구를 대기시키고 종료하는 제 3단계와;

상기 제 2단계에서 거짓인 경우에는 사용자의 수신 요구를 만족시켜 준 후, 수신이 끝난 버퍼는 해제하고, 해제한 버퍼는 보드 수신부(350)의 "프리버퍼큐(30)"에 되돌려 주는 제 4단계와;

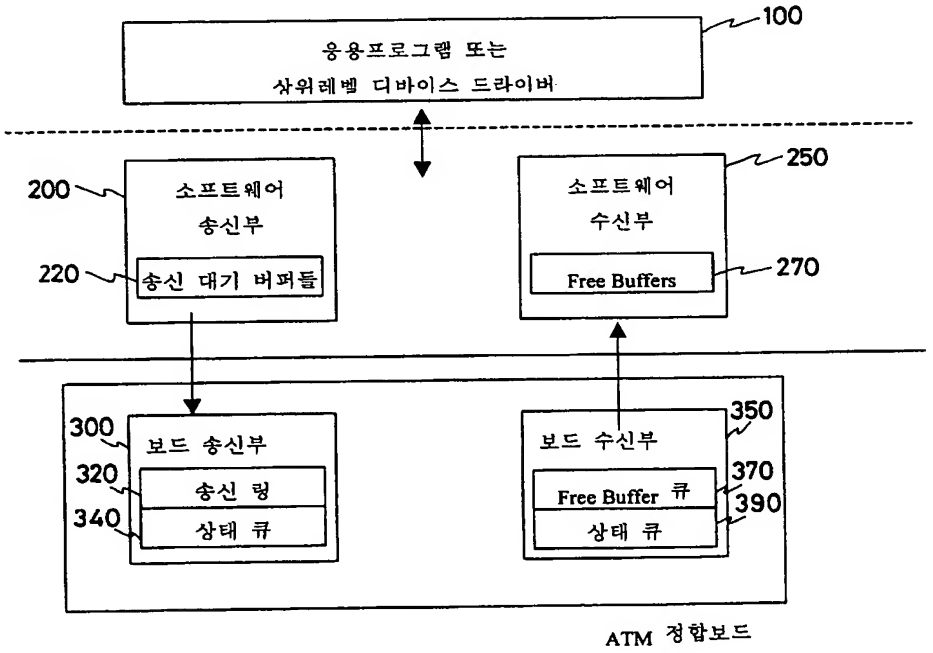
해당 연결의 현재 각 연결의 버퍼하한을 초과하여 할당한 버퍼의 개수가 '0'보다 큰지를 판별하는 제 5단계와;

상기 제 5단계에서 참인 경우에는 전체 버퍼 상한의 나머지 50%의 개수중 현재 사용 가능한 버퍼의 개수를 수신이 끝난 버퍼의 개수만큼 늘이고, 해당 연결의 현재 각 연결의 버퍼 하한을 초과하여 할당한 버퍼 수를 수신이 끝난 버퍼의 개수만큼 줄이는 제 7단계와;

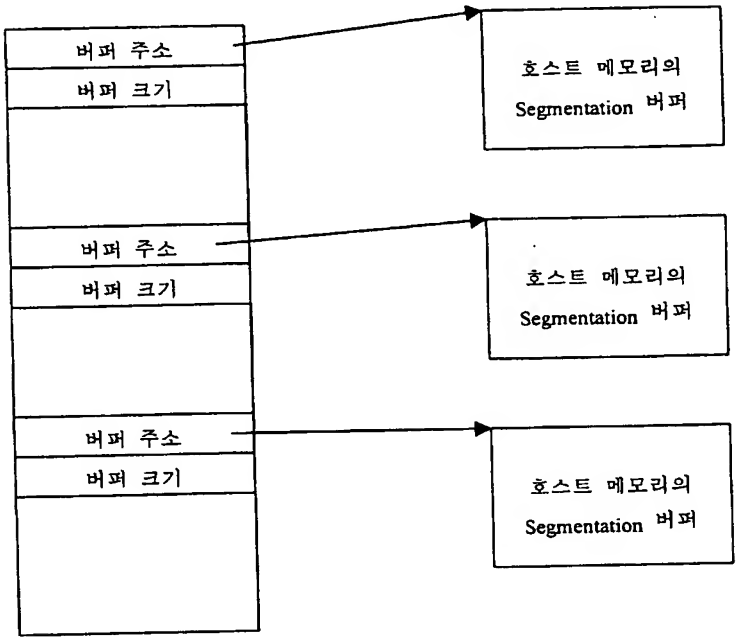
상기 제 5단계에서 거짓인 경우와 제 7단계의 수행이 끝난 경우에는 해당 연결의 사용자가 수신하기를 기다리는 버퍼 수를 수신이 끝난 버퍼의 개수만큼 줄이는 제 8단계를 포함하는 것을 특징으로 하는 버퍼제어 방법

도면

도면 1



도면 2

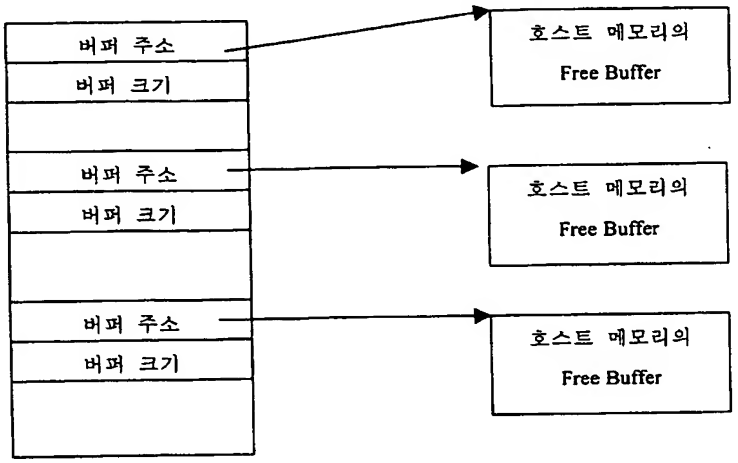


송신 링(320)

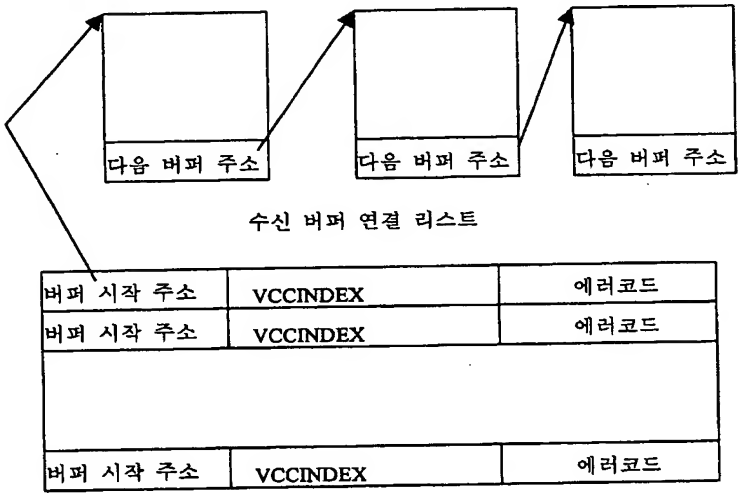
버퍼 시작 주소	Segmentation 결과	에러코드
버퍼 시작 주소	Segmentation 결과	에러코드
버퍼 시작 주소	Segmentation 결과	에러코드

송신 상태 큐(340)

도면 3

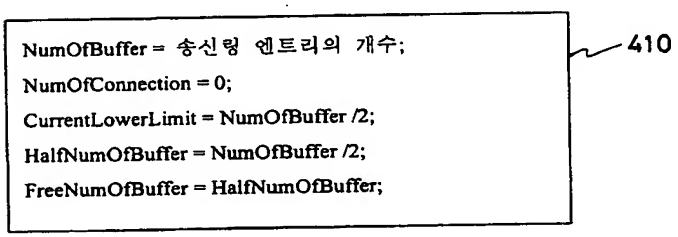


Free Buffer 큐 (370)

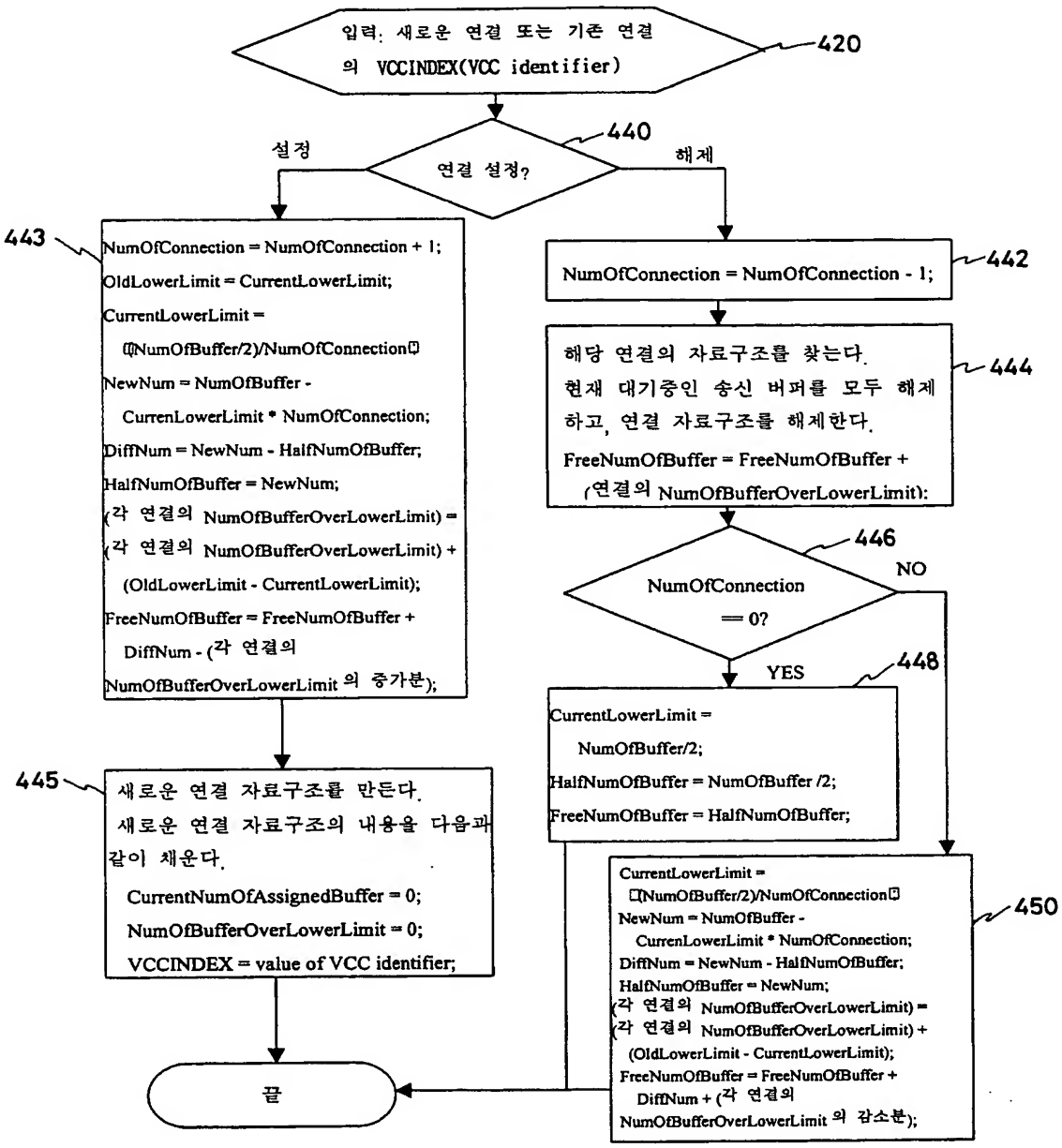


수신 상태 큐 (390)

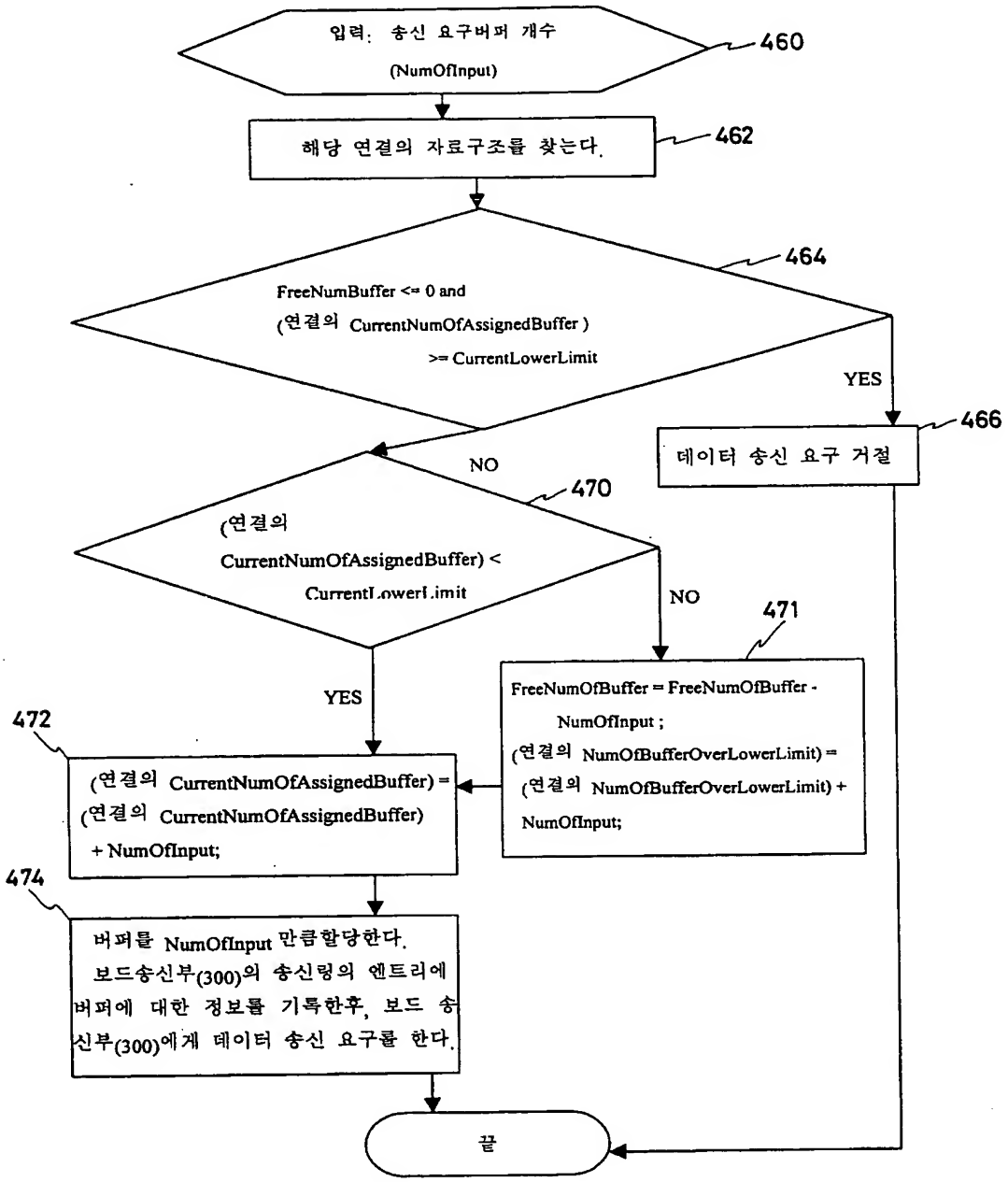
도면 4a



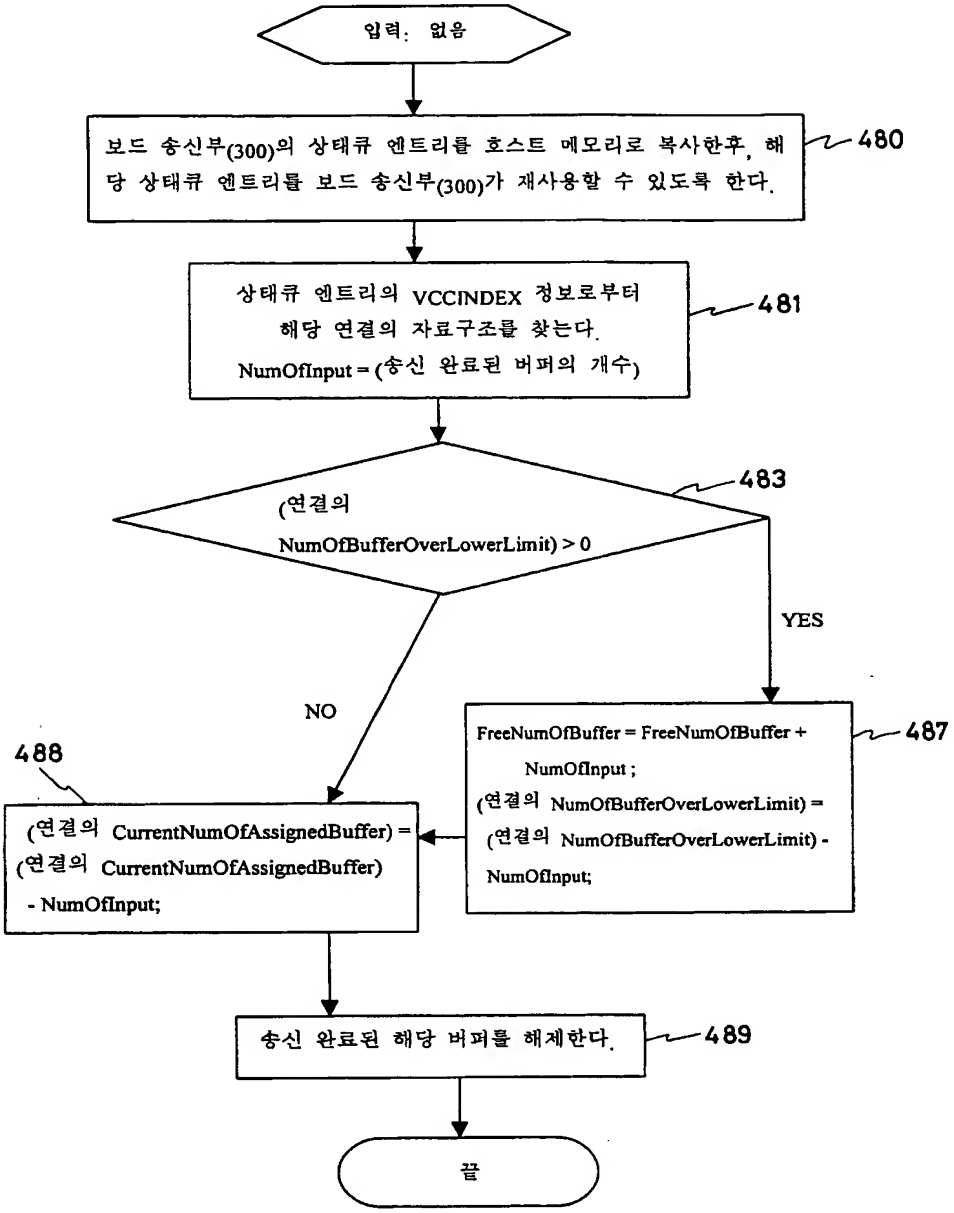
도면 4b


입력: 새로운 연결 또는 기존 연결의 VCCINDEX(VCC identifier)
420
연결 설정?
440
설정
해제
443
NumOfConnection = NumOfConnection + 1;
OldLowerLimit = CurrentLowerLimit;
CurrentLowerLimit = ⌊NumOfBuffer/2)/NumOfConnection⌋
NewNum = NumOfBuffer - CurrenLowerLimit * NumOfConnection;
DiffNum = NewNum - HalfNumOfBuffer;
HalfNumOfBuffer = NewNum;
(각 연결의 NumOfBufferOverLowerLimit) = (각 연결의 NumOfBufferOverLowerLimit) + (OldLowerLimit - CurrentLowerLimit);
FreeNumOfBuffer = FreeNumOfBuffer + DiffNum - (각 연결의 NumOfBufferOverLowerLimit 의 증가분);
445
새로운 연결 자료구조를 만든다.
새로운 연결 자료구조의 내용을 다음과 같이 채운다.
CurrentNumOfAssignedBuffer = 0;
NumOfBufferOverLowerLimit = 0;
VCCINDEX = value of VCC identifier;
끝
NumOfConnection = NumOfConnection - 1;
442
해당 연결의 자료구조를 찾는다.
현재 대기중인 송신 버퍼를 모두 해제하고, 연결 자료구조를 해제한다.
444
FreeNumOfBuffer = FreeNumOfBuffer + (연결의 NumOfBufferOverLowerLimit);
446
NumOfConnection == 0?
NO
YES
448
CurrentLowerLimit = NumOfBuffer/2;
HalfNumOfBuffer = NumOfBuffer /2;
FreeNumOfBuffer = HalfNumOfBuffer;
CurrentLowerLimit = ⌊NumOfBuffer/2)/NumOfConnection⌋
NewNum = NumOfBuffer - CurrenLowerLimit * NumOfConnection;
DiffNum = NewNum - HalfNumOfBuffer;
HalfNumOfBuffer = NewNum;
(각 연결의 NumOfBufferOverLowerLimit) = (각 연결의 NumOfBufferOverLowerLimit) + (OldLowerLimit - CurrentLowerLimit);
FreeNumOfBuffer = FreeNumOfBuffer + DiffNum + (각 연결의 NumOfBufferOverLowerLimit 의 감소분);
450

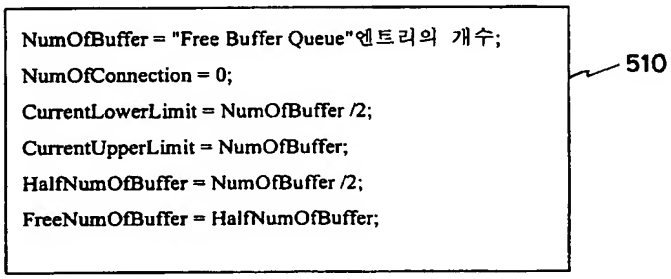
도면 4c


입력: 송신 요구버퍼 개수
(NumOfInput)
460
해당 연결의 자료구조를 찾는다.
462
FreeNumBuffer <= 0 and
(연결의 CurrentNumOfAssignedBuffer)
>= CurrentLowerLimit
464
YES
데이터 송신 요구 거절
466
NO
(연결의
CurrentNumOfAssignedBuffer) <
CurrentLowerLimit
470
NO
YES
FreeNumOfBuffer = FreeNumOfBuffer -
NumOfInput ;
(연결의 NumOfBufferOverLowerLimit) =
(연결의 NumOfBufferOverLowerLimit) +
NumOfInput;
471
(연결의 CurrentNumOfAssignedBuffer) =
(연결의 CurrentNumOfAssignedBuffer)
+ NumOfInput;
472
버퍼를 NumOfInput 만큼할당한다.
보드송신부(300)의 송신링의 엔트리에
버퍼에 대한 정보를 기록한후, 보드 송
신부(300)에게 데이터 송신 요구를 한다.
474
끝

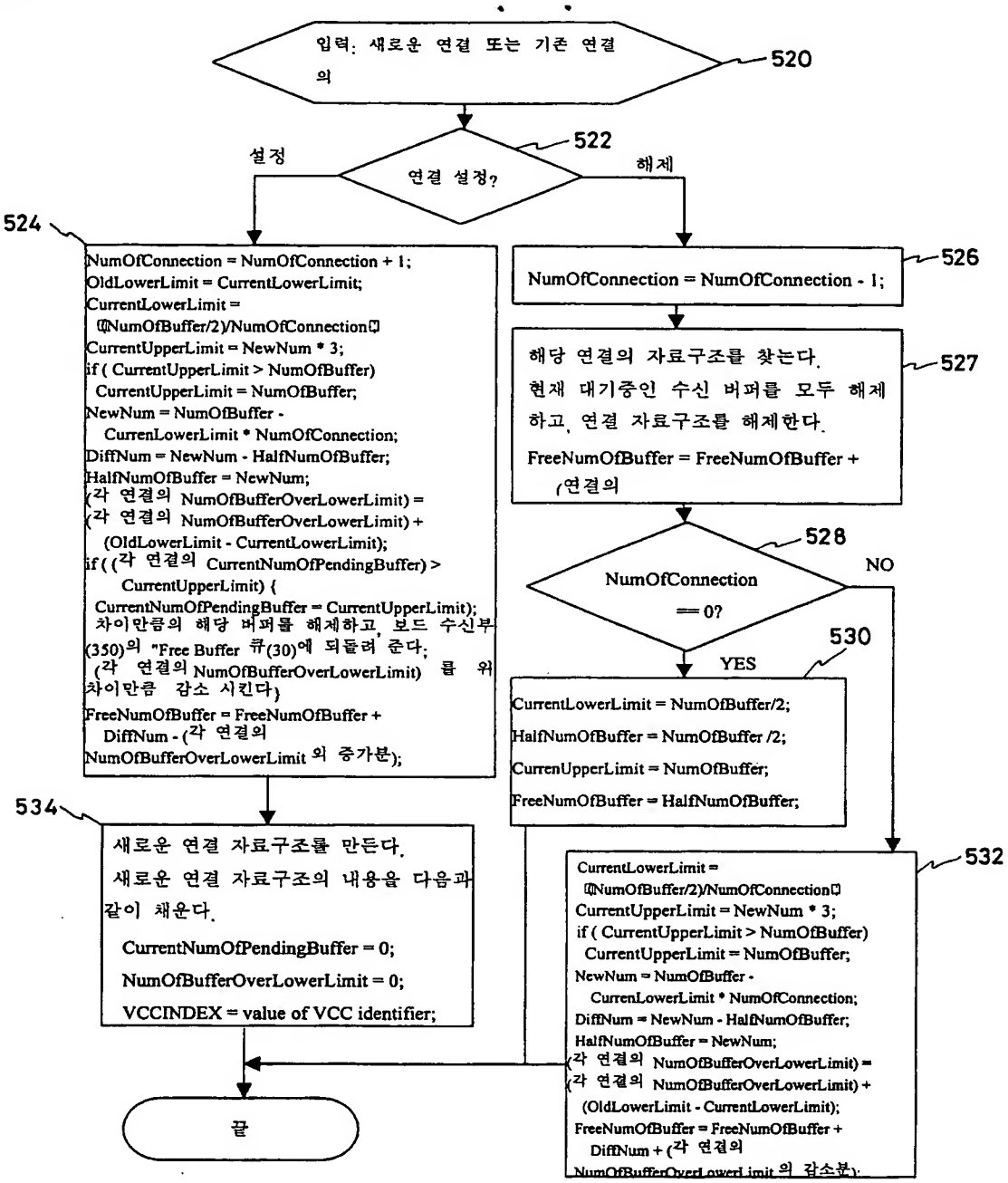
도면 4d


입력: 없음
보드 송신부(300)의 상태큐 엔트리를 호스트 메모리로 복사한후, 해당 상태큐 엔트리를 보드 송신부(300)가 재사용할 수 있도록 한다.
480
상태큐 엔트리의 VCCINDEX 정보로부터 해당 연결의 자료구조를 찾는다.
NumOfInput = (송신 완료된 버퍼의 개수)
481
(연결의 NumOfBufferOverLowerLimit) > 0
483
YES
NO
FreeNumOfBuffer = FreeNumOfBuffer + NumOfInput ;
(연결의 NumOfBufferOverLowerLimit) = (연결의 NumOfBufferOverLowerLimit) - NumOfInput;
487
488
(연결의 CurrentNumOfAssignedBuffer) = (연결의 CurrentNumOfAssignedBuffer) - NumOfInput;
송신 완료된 해당 버퍼를 해제한다.
489
끝

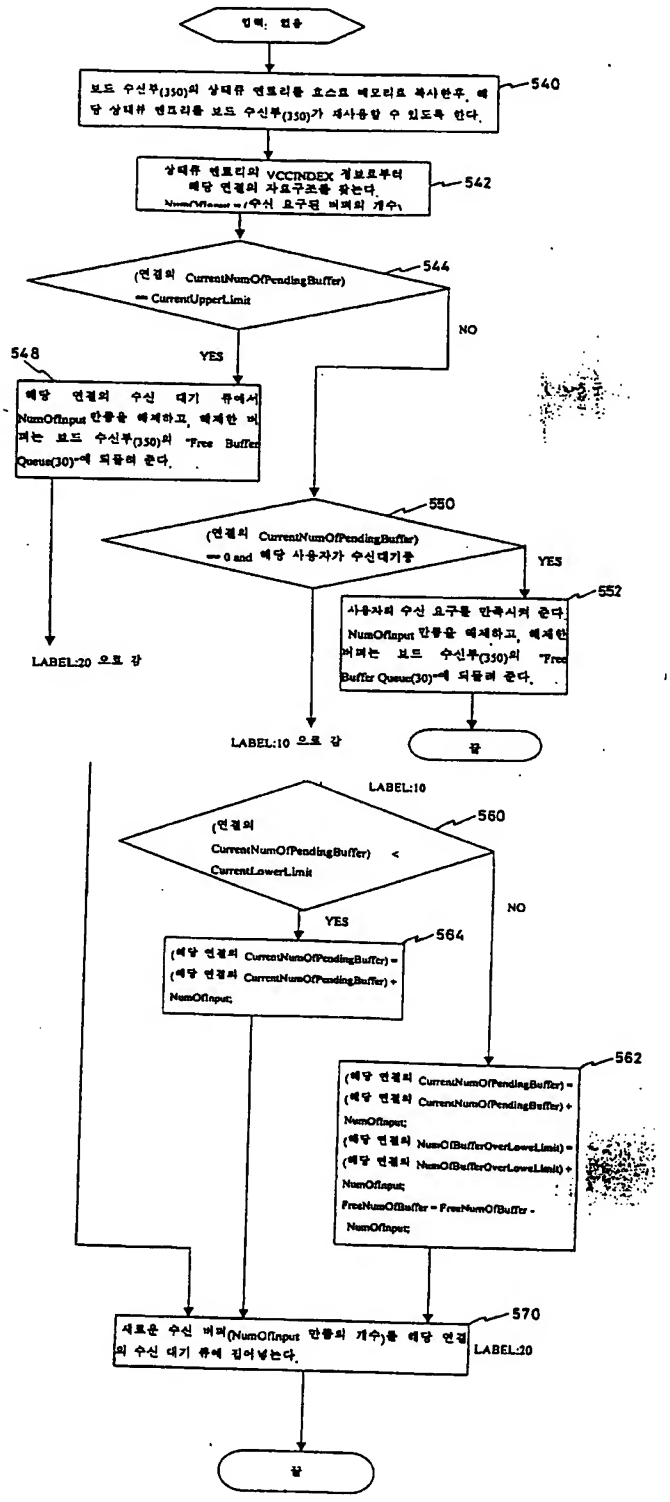
도면 5a


NumOfBuffer = "Free Buffer Queue"엔트리의 개수;
NumOfConnection = 0;
CurrentLowerLimit = NumOfBuffer /2;
CurrentUpperLimit = NumOfBuffer;
HalfNumOfBuffer = NumOfBuffer /2;
FreeNumOfBuffer = HalfNumOfBuffer;
510

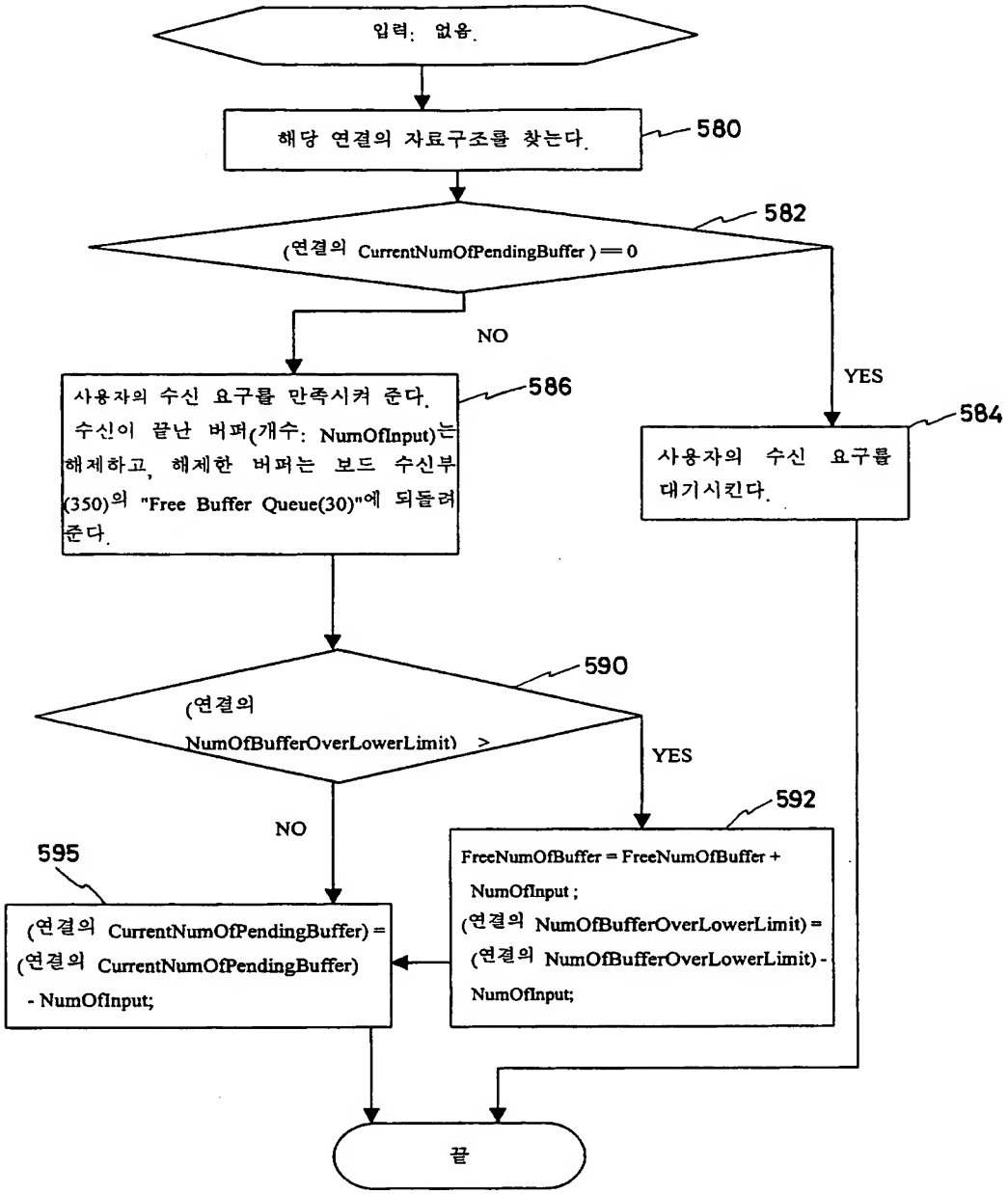
도면 5b


입력: 새로운 연결 또는 기존 연결의
520
연결 설정?
522
설정
해제
524
NumOfConnection = NumOfConnection + 1;
OldLowerLimit = CurrentLowerLimit;
CurrentLowerLimit = ⌈NumOfBuffer/2)/NumOfConnection⌋
CurrentUpperLimit = NewNum * 3;
if (CurrentUpperLimit > NumOfBuffer)
CurrentUpperLimit = NumOfBuffer;
NewNum = NumOfBuffer - CurrenLowerLimit * NumOfConnection;
DiffNum = NewNum - HalfNumOfBuffer;
HalfNumOfBuffer = NewNum;
(각 연결의 NumOfBufferOverLowerLimit) = (각 연결의 NumOfBufferOverLowerLimit) + (OldLowerLimit - CurrentLowerLimit);
if ((각 연결의 CurrentNumOfPendingBuffer) > CurrentUpperLimit) {
CurrentNumOfPendingBuffer = CurrentUpperLimit;
차이만큼의 해당 버퍼를 해제하고, 보드 수신부(350)의 "Free Buffer 큐(30)에 되돌려 준다;
(각 연결의 NumOfBufferOverLowerLimit) 를 위 차이만큼 감소 시킨다}
FreeNumOfBuffer = FreeNumOfBuffer + DiffNum - (각 연결의 NumOfBufferOverLowerLimit 의 증가분);
526
NumOfConnection = NumOfConnection - 1;
527
해당 연결의 자료구조를 찾는다.
현재 대기중인 수신 버퍼를 모두 해제하고, 연결 자료구조를 해제한다.
FreeNumOfBuffer = FreeNumOfBuffer + (연결의
528
NumOfConnection == 0?
NO
YES
530
CurrentLowerLimit = NumOfBuffer/2;
HalfNumOfBuffer = NumOfBuffer /2;
CurrenUpperLimit = NumOfBuffer;
FreeNumOfBuffer = HalfNumOfBuffer;
532
CurrentLowerLimit = ⌈NumOfBuffer/2)/NumOfConnection⌋
CurrentUpperLimit = NewNum * 3;
if (CurrentUpperLimit > NumOfBuffer)
CurrentUpperLimit = NumOfBuffer;
NewNum = NumOfBuffer - CurrenLowerLimit * NumOfConnection;
DiffNum = NewNum - HalfNumOfBuffer;
HalfNumOfBuffer = NewNum;
(각 연결의 NumOfBufferOverLowerLimit) = (각 연결의 NumOfBufferOverLowerLimit) + (OldLowerLimit - CurrentLowerLimit);
FreeNumOfBuffer = FreeNumOfBuffer + DiffNum + (각 연결의 NumOfBufferOverLowerLimit 의 감소분);
534
새로운 연결 자료구조를 만든다.
새로운 연결 자료구조의 내용을 다음과 같이 채운다.
CurrentNumOfPendingBuffer = 0;
NumOfBufferOverLowerLimit = 0;
VCCINDEX = value of VCC identifier;
끝

도면 5c


입력: 없음
보드 수신부(350)의 상태류 엔트리를 호스트 메모리로 복사한후, 해당 상태류 엔트리를 보드 수신부(350)가 재사용할 수 있도록 한다.
540
상태류 엔트리의 VCCINDEX 정보로부터 해당 연결의 자료구조를 얻는다.
542
(연결의 CurrentNumOfPendingBuffer) == CurrentUpperLimit
544
YES
NO
548
해당 연결의 수신 대기 큐에서 NumOfInput 만큼을 제거하고, 제거한 버퍼는 보드 수신부(350)의 "Free Buffer Queue(30)"에 되돌려 준다.
LABEL:20 으로 감
550
(연결의 CurrentNumOfPendingBuffer) == 0 and 해당 사용자가 수신대기중
YES
552
사용자의 수신 요구를 만족시켜 준다. NumOfInput 만큼을 제거하고, 제거한 버퍼는 보드 수신부(350)의 "Free Buffer Queue(30)"에 되돌려 준다.
끝
LABEL:10 으로 감
LABEL:10
(연결의 CurrentNumOfPendingBuffer) < CurrentLowerLimit
560
YES
NO
564
(해당 연결의 CurrentNumOfPendingBuffer) = (해당 연결의 CurrentNumOfPendingBuffer) + NumOfInput;
562
(해당 연결의 CurrentNumOfPendingBuffer) = (해당 연결의 CurrentNumOfPendingBuffer) + NumOfInput;
(해당 연결의 NumOfBufferOverLowerLimit) = (해당 연결의 NumOfBufferOverLowerLimit) + NumOfInput;
FreeNumOfBuffer = FreeNumOfBuffer - NumOfInput;
새로운 수신 버퍼(NumOfInput 만큼의 개수)를 해당 연결의 수신 대기 큐에 집어넣는다.
570
LABEL:20
끝

도면 5d


입력: 없음.
해당 연결의 자료구조를 찾는다.
580
(연결의 CurrentNumOfPendingBuffer) == 0
582
NO
YES
사용자의 수신 요구를 만족시켜 준다. 수신이 끝난 버퍼(개수: NumOfInput)는 해제하고, 해제한 버퍼는 보드 수신부(350)의 "Free Buffer Queue(30)"에 되돌려 준다.
586
사용자의 수신 요구를 대기시킨다.
584
(연결의 NumOfBufferOverLowerLimit) >
590
YES
NO
FreeNumOfBuffer = FreeNumOfBuffer + NumOfInput ;
(연결의 NumOfBufferOverLowerLimit) = (연결의 NumOfBufferOverLowerLimit) - NumOfInput;
592
595
(연결의 CurrentNumOfPendingBuffer) = (연결의 CurrentNumOfPendingBuffer) - NumOfInput;
끝